The University of Miami Wave Model (UMWM)
Version 1.0.0

Description and User's Manual

M.A. Donelan and M. Curcic

April, 2012

Rosenstiel School of Marine and Atmospheric Science
University of Miami

Contents

1 Model description **2**

2 Source functions **2**
2.1 Wind input function S_{in} . . . 3
2.2 Wave dissipation function S_{ds} . . . 3
2.3 Dissipation by turbulence source function S_{ds} . . . 4
2.4 The non-linear wave-wave interaction function S_{nl} . . . 4
2.5 The bottom friction source function S_{bp} . . . 4
2.6 The bottom percolation source function S_{bp} . . . 4

3 Numerical approaches **5**
3.1 Spatial discretization . . . 5
3.2 Time discretization . . . 6

4 Stress calculation **7**

5 Software implementation **8**

6 Installation and setup **9**
6.1 Obtaining the software package . . . 9
6.2 Compiling the source code . . . 9
6.3 Setting up the simulation . . . 10

7 Running the model **13**
7.1 Restarting the model . . . 13

8 Reading model output data **13**

9 References **14**

1 Model description

The University of Miami Wave Model (UMWM) is a prediction model for wave energy and wind stress on the interface between a liquid and a gas. It proceeds through a numerical solution of the wave energy balance equation on a horizontal 2-dimensional grid. The wave energy is a positive definite quantity in logarithmically spaced frequency (f) bins and uniformly spaced directional (ϕ) bins. The energy spectrum is carried as the wavenumber-directional surface elevation variance spectrum and every frequency at each location is identified with the theoretical wavenumber. Thus wave energy is a 5-dimensional quantity (x, y, k, ϕ, t). Time (t) evolution is done by integrating in time the energy balance equation:

$$\frac{\partial E'}{\partial t} + \frac{\partial(\dot{\mathbf{x}}E')}{\partial \mathbf{x}} + \frac{\partial(\dot{k}E')}{\partial k} + \frac{\partial(\dot{\phi}E')}{\partial \phi} = \rho_w g \sum_{i=1}^{n} S_i \tag{1.1}$$

where $E'(\mathbf{x}, k, \phi, t)$ is the energy spectrum, and $\dot{\mathbf{x}} = \mathbf{c}_g + \mathbf{u}$, where $\mathbf{c}_g$ is the group velocity and $\mathbf{u}$ is the current in the wave boundary layer. The second, third and fourth terms on the left-hand side are the advection terms in geographical, wavenumber and directional space, respectively. Advection in wavenumber space is non-zero only in case of changing ocean currents or moving bottom. Here, currents are considered to be quasi-stationary on the time scale of wave growth/decay. Thus, this term is neglected. Advection in directional space (refraction) is non-zero in case of variable water depth and/or inhomogeneous currents. $\sum_{i=1}^{n} S_i$ are the source/sink functions that act to grow/decay the waves locally. ρ_w is the liquid density and g is acceleration due to gravity.

The energy spectrum and the variance spectrum are related by:

$$E'(\mathbf{x}, k, \phi) = \rho_w g E(\mathbf{x}, k, \phi) \tag{1.2}$$

For convenience, the variance spectrum is the predicted quantity.

2 Source functions

The source functions S are parametric descriptions of the various phenomena that increase, decrease or interchange among wavenumbers the energy in the wavenumber spectrum. The wavenumber spectrum is evaluated in separable magnitude and direction bins. The phenomena relevant to the prediction of storm waves in water of arbitrary depth are:

(i) Input of energy and momentum from the wind and export of wave energy and momentum to the wind when the waves overrun or run against the wind;

(ii) Dissipation of wave energy at and near the surface due to viscosity, ambient turbulence and breaking;

(iii) Enhanced dissipation at and near the top and bottom interfaces due to shoaling;

(iv) Movement of energy to lower wavenumbers (down-shifting) due to nonlinear interactions including breaking;

(v) Enhanced dissipation due to straining by longer waves.

The theoretical and experimental justifications for the source functions corresponding to these phenomena are given in Donelan et al. (2012). Here the source functions are simply listed.

2.1 Wind input function S_{in}

Jeffreys' (1924, 1925) sheltering hypothesis leads to S_{in} of the form:

$$S_{in} = A_1 \left(U_{\lambda/2}\cos\theta - c - u\cos\phi - v\sin\phi\right)\left|U_{\lambda/2}\cos\theta - c - u\cos\phi - v\sin\phi\right| \frac{k\omega}{g}\frac{\rho_a}{\rho_w}E(k,\phi) \quad (2.1)$$

where θ is the angle between wind direction, ψ, and waves of wavenumber, k and direction, ϕ. A_1 is the sheltering coefficient.

S_{in} is positive (energy and momentum transferred from wind to waves) when $U_{\lambda/2}\cos\theta > c + u\cos\phi + v\sin\phi$, and negative (energy and momentum transferred from waves (swell) to wind) when $0 < U_{\lambda/2}\cos\theta < c + u\cos\phi + v\sin\phi$ or when the waves (swell) propagate against the wind, $\cos\phi < 0$. As waves approach full development S_{in} goes to zero; i.e. the direct wind forcing vanishes. The sheltering coefficient, which describes the strength of the source/sink function, is different depending on whether S_{in} is positive (wind sea) or negative, when the waves run before the wind or against it (swell). The wind velocity is that at one half wavelength above the surface up to the top of the logarithmic layer, which is usually taken to be 20 m in the field.

$$A_1 = \begin{cases} 0.11, & \text{if } U_{\lambda/2}\cos\theta > c + u\cos\phi + v\sin\phi, \text{ wind sea} \\ 0.01, & \text{if } 0 < U_{\lambda/2}\cos\theta < c + u\cos\phi + v\sin\phi, \text{ swell with wind} \\ 0.1, & \text{if } \cos\phi < 0, \text{ swell against wind} \end{cases} \quad (2.2)$$

2.2 Wave dissipation function S_{ds}

The wave breaking dissipation source function is strongly nonlinear in the saturation spectrum, $B(k,\phi) = k^4 F(k,\phi)$. In addition the dissipation is enhanced by the straining due to the velocity field of all longer waves and by the plunging breakers that occur for small values of kd (wavenumber times depth). Finally viscosity acts preferentially on short waves to dissipate them.

$$S_{ds}(k,\phi) = -A_2 \coth(kd)\left[\left(1 + A_3\overline{\chi^2}(k,\phi)\right)^2 B(k,\phi)\right]^{\frac{5}{2}} \omega E(k,\phi) - 4\nu k^2 E(k,\phi) \quad (2.3)$$

where $\overline{\chi^2}(k,\phi)$ is the sum-squared slope in direction ϕ of all waves longer than $\lambda(k)$ and ν is the kinematic viscosity of the liquid. Coefficients A_2 and A_3 have values 42 and 120, respectively.

2.3 Dissipation by turbulence source function S_{ds}

Turbulence mixing in the wave boundary layer attenuates waves. The wave dissipation due to ambient turbulence is:

$$S_{dt} = -A_4 u_{*w} k E(k,\phi) \tag{2.4}$$

where u_{*w} is the friction velocity in the water near the surface and $A_4 = 0.01$.

2.4 The non-linear wave-wave interaction function S_{nl}

A quantity proportional to the energy dissipated in spilling is passed to longer waves in the next two lower frequency bins. The amount transferred decays exponentially with the square of the relative frequency separation:

$$S_{nl}(k,\phi) = A_5 \left[b_1 S_{sb}(k - \Delta k, \phi) + b_2 S_{sb}(k - 2\Delta k, \phi) - S_{sb}(k,\phi) \right] \tag{2.5}$$

where $A_5 = 5$, $b_1 = \exp(-16(\frac{\Delta f}{f})^2)$, $b_2 = \exp(-16(2\frac{\Delta f}{f})^2)$, and b_1 and b_2 are normalized such that $b_1 + b_2 = 1$.

2.5 The bottom friction source function S_{bp}

The bottom friction source function is related to orbital velocity at the bottom and the roughness of the bed. Komen et al. (1994) give the following form for a sandy bed:

$$S_{bf} = -G_f \frac{k}{\sinh 2kd} E(k,\phi) \tag{2.6}$$

where the roughness factor, G_f varies from 0.001 to 0.01 m/s, depending on bed roughness.

2.6 The bottom percolation source function S_{bp}

On a porous bed the percolation of flow through the bed induces wave energy dissipation. The dissipation rate due to percolation is given by Shemdin et al. (1978):

$$S_{bp} = -G_p \frac{k}{\cosh^2 kd} E(k,\phi) \tag{2.7}$$

where the permeability factor, G_p varies from 0.0006 to 0.01 m/s, depending on sand grain size.

3 Numerical approaches

3.1 Spatial discretization

The time evolution of the variance spectrum due to advection in Cartesian projection is given by:

$$\frac{\partial E}{\partial t} = -\frac{\partial[(c_g \cos\phi + u)E]}{\partial x} - \frac{\partial[(c_g \sin\phi + v)E]}{\partial y} - \frac{\partial(\dot{\phi}E)}{\partial \phi} \tag{3.1}$$

where u and v are ocean current components in x and y, respectively, and $\dot{\phi}$ is the rotation rate. Both geographical propagation and refraction terms are discretized using first order upstream differencing. This scheme is positive-definite, quantity conserving, implicitly diffusive and computationally efficient. A certain amount of diffusion is desirable in order to avoid swell separation between discrete directional and frequency bins. A spatial differencing operator is then discretized as:

$$\frac{\partial(\dot{x}E)}{\partial x} \approx \frac{\Phi_{i+1/2} - \Phi_{i-1/2}}{\Delta x} \tag{3.2}$$

where i is a discrete index along dimension x. Fluxes at cell edges $\Phi_{i+1/2}$ and $\Phi_{i-1/2}$ are defined as:

$$\Phi_{i+1/2} = \frac{\dot{x}_{i+1/2} + |\dot{x}_{i+1/2}|}{2} E_i + \frac{\dot{x}_{i+1/2} - |\dot{x}_{i+1/2}|}{2} E_{i+1} \tag{3.3}$$

$$\Phi_{i-1/2} = \frac{\dot{x}_{i-1/2} + |\dot{x}_{i-1/2}|}{2} E_{i-1} + \frac{\dot{x}_{i-1/2} - |\dot{x}_{i-1/2}|}{2} E_i \tag{3.4}$$

and

$$\dot{x}_{i\pm1/2} = \frac{\dot{x}_i + \dot{x}_{i\pm1}}{2} \tag{3.5}$$

The above treatment of flux differencing ensures upstream-definiteness. For propagation in two-dimensional space, the stability of the scheme is ensured for:

$$\mu = \frac{\dot{x}\Delta t}{\min(\Delta x, \Delta y)} < \frac{1}{\sqrt{2}} \tag{3.6}$$

where μ is the Courant number. Depending on the choice of number of directional bins, the stability criterion is more permissive:

$$\mu = \frac{\dot{x}\Delta t}{\min(\Delta x, \Delta y)} < \cos\left(\frac{\pi}{4} - \frac{\Delta\phi}{2}\right) \tag{3.7}$$

where $\Delta\phi$ is the directional bin size. To ensure that (3.7) holds, the number of directions must be divisible by 8.

For ocean grid cells next to the land, an open boundary condition is applied (energy can freely propagate into land). Same is applied for domain edges, except that there can be incoming wave energy from the boundaries, if provided by the user. In the case of global domain simulation, periodic boundary conditions are applied at East and West domain edges.
The rotation rate $\dot{\phi}$ in the refraction term is evaluated as:

$$\dot{\phi} = \frac{\partial(c\sin\phi + v)}{\partial x} - \frac{\partial(c\cos\phi + u)}{\partial y} \tag{3.8}$$

The change due to refraction is then computed using (3.2)-(3.5). Positive and negative values of $\dot{\phi}$ correspond to counter-clockwise and clockwise rotation of energy, respectively. The stability constraint for the refraction term is the same as for one-dimensional advection:

$$\mu = \frac{\dot{\phi}\Delta t}{\Delta\phi} < 1 \tag{3.9}$$

For most domain cells, the allowed refraction time step is larger than the advective step given in (3.7). In case that condition (3.9) is violated, which can occur on sharp bathymetric or current gradients, the rotation at these points is limited so that $\mu = 1$. This affects the solution insignificantly, while maintaining computational efficiency. Because the domain is periodic in directional space, there is no need for boundary conditions.

3.2 Time discretization

Once all the source terms in (1.1) have been evaluated, $E(k,\phi)$ is integrated forward in time. We evaluate the contribution from source and advection terms separately:

$$\frac{\partial E}{\partial t} = \left(\frac{\partial E}{\partial t}\right)_s + \left(\frac{\partial E}{\partial t}\right)_a \tag{3.10}$$

The contribution from source terms can be written as:

$$\left(\frac{\partial E}{\partial t}\right)_s = \sum_{i=1}^{n} S_i^* E \tag{3.11}$$

where S_i^* is just S_i/E. Then, by integrating (3.11) over a finite time interval Δt, a solution is available in the form of:

$$E_s^{n+1} = E^n exp\left(\sum_{i=1}^{n} S_i^* \Delta t\right) \tag{3.12}$$

The time increment Δt is dynamically computed so that the variance spectrum E can only grow by a pre-determined, finite factor:

$$\frac{E_s^{n+1}}{E^n} = exp\left(\sum_{i=1}^{n} S_i^* \Delta t\right) < r \tag{3.13}$$

where r is usually set between 1.5 and 2. Lower values of r will draw E closer to the solution attractor.

Then, a time-splitting approach is used to achieve a more stable integration:

$$E^* = \frac{E^n + E_s^{n+1}}{2} \tag{3.14}$$

E^* is used to compute the advection and refraction terms described in the previous section. Finally, their contribution is evaluated by simple forward-Euler differencing:

$$E^{n+1} = E_s^{n+1} - \Delta t \left[\frac{\partial[(c_g \cos\phi + u)E^*]}{\partial x} + \frac{\partial[(c_g \sin\phi + v)E^*]}{\partial y} + \frac{\partial(\dot{\phi}E^*)}{\partial \phi}\right] \tag{3.15}$$

The above approach is applied to the prognostic part of the spectrum. A cut-off frequency, f_c, which separates the prognostic and diagnostic parts, is proportional to the peak frequency of the fully-developed Pierson-Moskowitz spectrum (Pierson and Moskowitz, 1964):

$$f_c = 4 f_{PM} = \frac{0.52g}{U_{10}} \tag{3.16}$$

For all bins higher than f_c, the waves are assumed to be: in equilibrium with the wind – traveling in the wind direction – and their spectral densities are established from a balance of wind input and dissipation. This approach is justified by the presumption that the quasi-equilibrium range is wider in higher wind conditions.

4 Stress calculation

Energy E and momentum M in the wave field are related by the phase speed:

$$E = Mc \tag{4.1}$$

The form drag components of the wind on the waves τ_x and τ_y are calculated from the wind input source function:

$$\tau_x = \rho_w g \int_{-\pi}^{\pi} \int_{k_{min}}^{k_{max}} \frac{S_{in}}{c} \cos\phi \, k \, dk \, d\phi \tag{4.2}$$

$$\tau_y = \rho_w g \int_{-\pi}^{\pi} \int_{k_{min}}^{k_{max}} \frac{S_{in}}{c} \sin\phi \, k \, dk \, d\phi \tag{4.3}$$

The form drag coefficient, Cd_f is then calculated:

$$Cd_f = \frac{\sqrt{\tau_x^2 + \tau_y^2}}{\rho_a U_z^2} \tag{4.4}$$

where U_z is the wind speed at the measured or modeled height z. The form drag is in the wave direction ϕ. The skin drag coefficient Cd_s, in the absence of waves, is in the wind direction ψ, and is computed from Von Karman's relationship:

$$Cd_s = \frac{u_*^2}{U^2(z)} = \frac{\kappa^2}{\left(\log\left(\frac{z}{z_0}\right)\right)^2} \tag{4.5}$$

where u_* is the friction velocity, $U(z)$ is wind speed at height z and κ is the Von Karman's constant. The roughness length z_0 is determined by the scale of the molecular sublayer for smooth flow:

$$z_0 = 0.132 \frac{\nu_a}{u_*} \tag{4.6}$$

where ν_a is the kinematic viscosity of air. In the presence of waves Cd_s is reduced due to the sheltering of the surface in the lee of steep waves. The degree of sheltering of the skin is taken to be proportional to the ratio of total drag to skin drag and may be as large as 50% corresponding to full sheltering of the lee face of each wave. The algorithm is as follows:

$$Cd_s^{new} = \frac{Cd_s^{old}}{3}\left(1 + 2\frac{Cd_s^{old}}{Cd_s^{old} + Cd_f}\right) \tag{4.7}$$

In order to obtain correct stress magnitude, the high-frequency limit should be equal to or greather than 2 Hz. A linear, wind-speed dependent tail is appended to the spectrum beyond the high-frequency limit. Low-frequency limit should be set to be equal to or lower than the expected spectrum peak frequency.

These are the stress components of the wind on the surface. They are the mechanical couplers with the atmospheric model.

5 Software implementation

UMWM source code is written in standard Fortran 90 programming language. It is written in readable and transparent form (e.g. free of low-level instructions), and is well documented. The model may be defined on any structured, curvilinear grid. Most common applications are on a Cartesian

and spherical (latitude-longitude) grid. A choice is available between limited-area (regional) and global simulation. In the latter case, a periodic boundary condition in x-direction is applied automatically.
The user is encouraged to look into the code for information that is not provided in this document. The model is open-source and is licensed under a GPL General Public License Version 3, `http://www.gnu.org/copyleft/gpl.html`.

6 Installation and setup

In order to successfully convey a numerical simulation with UMWM, a procedure is usually as follows:

1) Obtaining the software package;

2) Compiling the source code;

3) Setting up the simulation;

4) Running the model;

5) Reading model output data.

In general, steps 1 and 2 need to be done only once. A working Fortran compiler must be provided by the user. Re-compiling the source code is necessary only if change has been made to any of the source files. Naturally, step 3 needs to be repeated if the setup of the experiment is to be changed. If only input forcing files are changed, one needs only to re-run the model (step 4).

6.1 Obtaining the software package

Current version of UMWM can be downloaded from this URL:

`http://rsmas.miami.edu/groups/umwm/download.html`

On a UNIX-like system (Linux, Mac OS X or other UNIX), unpack the downloaded file by typing `tar xzf umwm-1.x.x.tgz` (`1.x.x` should be replaced with the appropriate version number). This will create the directory `umwm`. In addition, UMWM requires NetCDF Fortran libraries for input and output. NetCDF libraries and installation instructions can be downloaded from `http://www.unidata.ucar.edu/downloa`
Also, a common build tool `make` or `gmake` is recommended for easier compilation. `make` is included by default in most UNIX/Linux distributions, but may need to be installed separately on a Microsoft Windows system.

6.2 Compiling the source code

Before doing a simulation, model code needs to be compiled. The source code is located in the `src/` directory. The user must edit the top section of the `Makefile` which contains rules about compiling the model:

```
# University of Miami Wave Model Makefile
#
# Compiler, flags, and libraty paths:
FC         = pgf90
FCFFLAGS   = -O3 -fastsse
CPPFLAGS   =
LD         = $(FC)
LDFLAGS    = $(FCFFLAGS)
NETCDF     = /usr/local
NETCDFPATH = -I$(NETCDF)/include -L$(NETCDF)/lib -lnetcdf
NETCDFLIB  = $(NETCDF)/lib/libnetcdf.a
```

Here, `FC` is the path to the Fortran compiler to be used. If the model is to be compiled in MPI mode, `FC` should be named appropriately (most commonly `mpif90`). `FCFLAGS` are the Fortran compiler flags to be used. UMWM does not require any special flags, so `FCFLAGS` are most likely to be optimization flags (optional). User should reference the manual pages for their compiler about the appropriate optimization flags. `CPPFLAGS` are the C-preprocessor flags (macros) that are optional for preprocessing the Fortran code. Only two macros are relevant for UMWM, `DMPI` and `DESMF`. `DMPI` is necessary when compiling the model in MPI mode. `DESMF` is used when model is to be used for coupling through the Earth System Modeling Framework (ESMF, `http://www.earthsystemmodeling.org`). Most likely, there is no need for the user to edit `LD` and `LDFLAGS`. `NETCDF` is the path to the directory where NetCDF libraries are installed. `NETCDFPATH` and `NETCDFLIB` should not be modified typically. Once `Makefile` is configured, type `make` to compile the code. The executable `umwm` will be generated and copied to the main model directory.

6.3 Setting up the simulation

Once the executable `umwm` has been generated, the user should set up a desired experiment. First step is to edit the main UMWM namelist that defines model parameters. The namelist is located in `namelists/main.nml`:

```
&DOMAIN
  isGlobal     = .FALSE.               ! Global (.TRUE.) or regional (.FALSE.)
  mm           = 50                    ! Domain size in x
  nm           = 11                    ! Domain size in y
  om           = 37                    ! Number of frequency bins
  pm           = 24                    ! Number of directions
  f1           = 0.0313                ! Lowest frequency bin [Hz]
  f2           = 2.0                   ! Highest frequency bin [Hz]
  startTimeStr = '2008-09-08_12:00:00' ! Simulation start time
  stopTimeStr  = '2008-09-08_14:00:00' ! Simulation end time
  dtg          = 3600                  ! Global (I/O) time step [s]
  restart      = .FALSE.               ! Restart from file
/
&PHYSICS
  g          = 9.80665 ! Gravitational acceleration   [m/s^2]
  nu         = 0.9E-6  ! Kinematic viscosity of water [m^2/s]
  sfct       = 0.07    ! Surface tension
```

```
  kappa       = 0.4     ! Von Karman constant
  z           = 10.     ! Height of the input wind speed [m]
  gustiness   = 0.0     ! Random wind gustiness factor (should be between 0 and 0.2)
  dmin        = 10.     ! Depth limiter [m]
  explim      = 0.7     ! Exponent limiter (0.69 ~ 100% growth)
  sin_fac     = 0.11    ! Input factor from following winds
  sin_diss1   = 0.10    ! Damping factor from opposing winds
  sin_diss2   = 0.01    ! Damping factor from swell overrunning wind
  sds_fac     = 42.     ! Breaking dissipation factor
  sds_power   = 2.5     ! Saturation spectrum power
  mss_fac     = 120.    ! Mean-square-slope adjustment to Sds
  snl_fac     = 5.      ! Wave energy downshifting factor
  sdt_fac     = 0.01    ! Dissipation due to turbulence factor
  sbf_fac     = 0.001   ! Bottom friction coefficient [m/s]
  sbp_fac     = 0.001   ! Bottom percolation coefficient [m/s]
/
&GRID
  gridid  = 1       ! Set 1 for Cartesian or 2 for spherical projection
  delx    = 12000   ! Grid spacing in x and y in: meters for gridid=1,
  dely    = 12000   ! degrees for gridid=2
  lat_s   = 51.5    ! Latitude of bottom left point; used for gridid=2
  lon_w   = -15.    ! Longitude of bottom left point; used for gridid=2
/
&INPUT_FLAGS        ! Set to 1 for input from file, or 0 for homogenous
                    ! fields input from &INPUT list below;
  dflag   = 0       ! Bathymetry
  excom   = 0       ! Set to 1 to exclude user chosen basins/lakes
  wflag   = 0       ! Winds
  pflag   = 0       ! Surface pressure
  tflag   = 0       ! Air temperature
  cflag   = 0       ! Currents
  rflag   = 0       ! Water density
/
&INPUT
  depth   = 1000    ! Water depth [m]
  u0      = 25      ! Wind magnitude [m/s]
  wdir0   = 0.      ! Wind direction [rad]
  sfcp0   = 101325. ! Surface pressure [Pa]
  temp0   = 300.    ! Surface air temperature [K]
  uc0     = 0.      ! x-component ocean current [m/s]
  vc0     = 0.      ! y-component ocean current [m/s]
  rhow0   = 1025.   ! Water density [kg/m^3]
/
&OUTPUT
  outgrid     = .TRUE.  ! Gridded output
  outspec     = .FALSE. ! Spectrum output
  xpl         =  45     ! Grid cell in x for stdout (screen)
  ypl         =   6     ! Grid cell in y for stdout (screen)
```

```
  restart_out = .FALSE. ! Restart output
/
```

Parameters in the DOMAIN list define domain size and start and end times of the simulation, and whether the model is to be run from a restart file. See next section about restarting the model. Parameters in the PHYSICS list will have effects on the physical results of the wave model. Height of the input wind speed z may need to be changed depending on the source of wind fields used as model forcing. The user may choose to set the gustiness parameter larger than zero. Since the wind is linearly interpolated in time between each input time interval dtg, this option may be used to add a stochastic component to wind fields when dtg is not short enough, e.g. 3 or 6 hours. Depth limiter dmin defines the smallest depth possible inside the model. Values lower than 10 meters are allowed, but may lead to a shorter time step and longer model integration time. explim is the exponential growth limiter used when integrating wave spectrum in time. Typically, it should be between 0.4 (50% growth) and 0.9 (140% growth). Higher values may lead to numerical oscillations and instability of the model. The parameters in INPUT_FLAGS list determine which fields are to be input from file. For fields that are not read from input files, homogeneous values are specified in INPUT list. If excom is set to 1, a namelist file namelists/exclude.nml is being read, and this file contains a list of model grid points that belong to enclosed seas that should be removed from the model domain. Finally, OUTPUT list specifies whether there will be gridded and spectrum output, and/or whether output of restart files is enabled. If restart_out is set to .TRUE., restart files will be written in the restart/ directory.

If any of outgrid or outspec in is set to .TRUE., desired output with the frequency of value of dtg will appear in the output/ directory. Gridded output will appear in the form:

umwmout_YYYY-MM-DD_hh:mm:ss.nc

where YYYY-MM-DD_hh:mm:ss denotes model time of the output. If outspec is set to .TRUE., a namelist file namelists/spectrum.nml is being read. An example spectrum namelist file may look like this:

```
LL
-89.66 25.89 NDBC42001
-93.67 25.79 NDBC42002
-96.66 25.96 NDBC42020
-94.41 29.23 NDBC42035
-88.80 30.09 NDBC42007
```

The first line of the file must contain either XY or LL. It describes whether the coordinates of spectrum output location are given in model grid indices or in longitude and latitude, respectively. In case of XY, spectrum at exact specified grid cell will be output. In case of LL, spectrum at grid cell nearest to specified location will be output. The following lines each contain 3 elements: x and y coordinate (or longitude and latitude) and location identifier (character string, up to 40 characters, no spaces). Spectrum output will appear in the form:

umwmspc_ID_YYYY-MM-DD_hh:mm:ss.nc

where ID is the location identifier specified by the user in namelists/spectrum.nml, and YYYY-MM-DD_hh:mm:ss is the initial time of the simulation. All output times during the simulation are stored in the same file.

Finally, parameters `xpl` and `ypl` specify the model grid location for the simple output to screen - time, time step, significant wave height, mean wave period and drag coefficient.

After configuring the namelists, user must provide the grid definition file. This file must be written in NetCDF form and placed in:

`input/umwm.gridtopo`

The `umwm.gridtopo` file must contain 3 two-dimensional fields: "lon", "lat" and "z". Fields "lon" and "lat" will define the domain for the simulation. Field "z" is the topography field where the 0-value is centered on mean sea-level, with positive z-axis upward (positive on land and negative in the ocean). The model will read this field, define seamask, discard the land values and apply the depth limiter `dmin`.

Optionally, the user may provide forcing files for wind, sea-level pressure, air temperature, surface ocean currents and water density fields. Input files must be written in NetCDF form and placed in:

`input/umwmin_YYYY-MM-DD_hh:mm:ss.nc`

These files will be read if any of the `wflag`, `pflag`, `tflag` or `cflag` are set to 1, and the files must contain following respective two-dimensional fields: "U10" and "V10", "PSFC", "T2", "UC" and "VC".

7 Running the model

Once namelists have been configured and input files are provided, the user can start the simulation by executing `umwm`. If the model was compiled in the MPI mode, refer to the documentation of the MPI implementation about executing programs in parallel. On a UNIX/Linux system with a common modern MPI implementation (MPICH2 or OpenMPI), one would type:

```
mpiexec -n 16 umwm
```

This command would execute `umwm` on 16 parallel processes.

7.1 Restarting the model

If `restart` is set to `.TRUE.` in `namelists/main.nml`, the model will read initial conditions from a restart file. `startTimeStr` must match the time string in the restart file name. For example, if a user specifies `'2012-09-08_18:00:00'` as `startTimeStr`, `umwmrst_2008-09-08_18:00:00.nc` file must be present in the `restart/` directory.

8 Reading model output data

As mentioned in the previous section, the output files will appear in the `output/` directory. The user can open and read output files with any program in a language that has NetCDF libraries provided - Fortran, MATLAB, Python, C/C++, Java etc. In addition, convenience NetCDF viewing programs exist, e.g. Ncview - `http://meteora.ucsd.edu/~pierce/ncview_home_page.html`. Sample scripts

to read gridded and spectrum output files from UMWM will be available from the model webpage: `http://rsmas.miami.edu/groups/umwm`.

9 References

Donelan, M.A., M. Curcic, S.S. Chen, and A.K. Magnusson, 2012: Modeling waves and wind stress. *Submitted to J. Geophys. Res.*

Jeffreys, H., 1924: On the formation of waves by wind. *Proc. Roy. Soc. A*, 107, pp. 189-206.

Jeffreys, H., 1925: On the formation of waves by wind, II. *Proc. Roy. Soc. A*, 110, pp. 341-347.

Komen, G. J., L. Cavaleri, M. Donelan, K. Hasselmann, S. Hasselmann, and P. A. E. M. Janssen, 1994: *Dynamics and Modelling of Ocean Waves.* Cambridge University Press, 532 pp.

Pierson, W. L. and L. Moskowitz, 1964: A proposed spectral form for fully developed wind seas based on the similarity theory of S. A. Kitaigorodskii. *J. Geophys. Res.*, 69, pp. 5181-5190.

Shemdin, O.H., K. Hasselmann, S.V. Hsiao, and K. Herterich, 1978: Non-linear and linear bottom interaction effects in shallow water, *Turbulent fluxes through the sea surface, wave dynamics and prediction,* A. Favre and K. Hasselmann (eds), Plenum, New York, pp. 347-372.